

UNCLASSIFIED

AD

AD-E403 688

Technical Report ARWSE-TR-14025

## **CODE SPEED MEASURING FOR VC++**

Tom Nealis

October 2015



**U.S. ARMY ARMAMENT RESEARCH, DEVELOPMENT AND  
ENGINEERING CENTER**

**Weapons and Software Engineering Center**

**Picatinny Arsenal, New Jersey**

Approved for public release; distribution is unlimited.

## UNCLASSIFIED

The views, opinions, and/or findings contained in this report are those of the author(s) and should not be construed as an official Department of the Army position, policy, or decision, unless so designated by other documentation.

The citation in this report of the names of commercial firms or commercially available products or services does not constitute official endorsement by or approval of the U.S. Government.

Destroy this report when no longer needed by any method that will prevent disclosure of its contents or reconstruction of the document. Do not return to the originator.

## UNCLASSIFIED

REPORT DOCUMENTATION PAGE				Form Approved OMB No. 0704-01-0188	
<p>The public reporting burden for this collection of information is estimated to average 1 hour per response, including the time for reviewing instructions, searching existing data sources, gathering and maintaining the data needed, and completing and reviewing the collection of information. Send comments regarding this burden estimate or any other aspect of this collection of information, including suggestions for reducing the burden to Department of Defense, Washington Headquarters Services Directorate for Information Operations and Reports (0704-0188), 1215 Jefferson Davis Highway, Suite 1204, Arlington, VA 22202-4302. Respondents should be aware that notwithstanding any other provision of law, no person shall be subject to any penalty for failing to comply with a collection of information if it does not display a currently valid OMB control number.</p> <p><b>PLEASE DO NOT RETURN YOUR FORM TO THE ABOVE ADDRESS.</b></p>					
1. REPORT DATE (DD-MM-YYYY) October 2015		2. REPORT TYPE Final		3. DATES COVERED (From - To)	
4. TITLE AND SUBTITLE  CODE SPEED MEASURING FOR VC++				5a. CONTRACT NUMBER	
				5b. GRANT NUMBER	
				5c. PROGRAM ELEMENT NUMBER	
6. AUTHORS  Tom Nealis				5d. PROJECT NUMBER	
				5e. TASK NUMBER	
				5f. WORK UNIT NUMBER	
7. PERFORMING ORGANIZATION NAME(S) AND ADDRESS(ES) U.S. Army ARDEC, WSEC Fire Control Systems & Technology Directorate (RDAR-WSF-M) Picatinny Arsenal, NJ 07806-5000				8. PERFORMING ORGANIZATION REPORT NUMBER	
9. SPONSORING/MONITORING AGENCY NAME(S) AND ADDRESS(ES) U.S. Army ARDEC, ESIC Knowledge & Process Management (RDAR-EIK) Picatinny Arsenal, NJ 07806-5000				10. SPONSOR/MONITOR'S ACRONYM(S)	
				11. SPONSOR/MONITOR'S REPORT NUMBER(S) Technical Report ARWSE-TR-14025	
12. DISTRIBUTION/AVAILABILITY STATEMENT  Approved for public release; distribution is unlimited.					
13. SUPPLEMENTARY NOTES					
14. ABSTRACT  It's often important to know how fast a snippet of code executes. This information allows the coder to make important decisions about the code that they create. Perhaps the coder is trying to find out what function call is taking so long or which code executes faster between different ways of accomplishing the same task. Regardless of the reason, faster code is typically more efficient and inherently will use less power.					
15. SUBJECT TERMS  QueryPerformanceCounter      QueryPerformanceFrequency					
16. SECURITY CLASSIFICATION OF:			17. LIMITATION OF ABSTRACT  SAR	18. NUMBER OF PAGES 9	19a. NAME OF RESPONSIBLE PERSON Tom Nealis
a. REPORT U	b. ABSTRACT U	c. THIS PAGE U			19b. TELEPHONE NUMBER (Include area code) (973) 724-8048



CONTENTS

	Page
Introduction	1
Methodology	1
Conclusions	3
Distribution List	5



# UNCLASSIFIED

## INTRODUCTION

For measuring code snippets on a windows machine with C++, it's convenient to use the built-in, high-resolution timestamps. For native code, the function `QueryPerformanceCounter` is available to retrieve this timestamp. There are other versions of this function for managed and kernel-mode code, but for the purposes of this discussion, the report will concentrate on native code.

## METHODOLOGY

In order to access this Application Program Interface (API), the coder is only required to include `windows.h`. The usage of this API can be easily understood from the following snippet of code taken from the Microsoft Developer Network (MSDN).

```
LARGE_INTEGER StartingTime, EndingTime, ElapsedMicroseconds;
LARGE_INTEGER Frequency;

QueryPerformanceFrequency(&Frequency);
QueryPerformanceCounter(&StartingTime);

// Activity to be timed

QueryPerformanceCounter(&EndingTime);
ElapsedMicroseconds.QuadPart = EndingTime.QuadPart - StartingTime.QuadPart;

//
// We now have the elapsed number of ticks, along with the
// number of ticks-per-second. We use these values
// to convert to the number of elapsed microseconds.
// To guard against loss-of-precision, we convert
// to microseconds *before* dividing by ticks-per-second.
//

ElapsedMicroseconds.QuadPart *= 1000000;
ElapsedMicroseconds.QuadPart /= Frequency.QuadPart;
```

`QueryPerformanceFrequency` must be called in order to get the performance-counter frequency. The frequency is returned in counts/ticks per second. This frequency does not change, so this function only needs to be called once. As the previous code shows, a timestamp is retrieved by calling `QueryPerformanceCounter` before and after the code to be measured.

Depending on the units desired, the value of the number of ticks is then multiplied by the appropriate value ( $10^6$  for  $\mu\text{s}$ ,  $10^3$  for ms, etc.). The previous example is 1000000 in order to get the value in  $\mu\text{s}$ . The result is then divided by the number of ticks per second.

As a simple example of this, consider the following program:

## UNCLASSIFIED

```
int _tmain(int argc, _TCHAR* argv[])
{
    LARGE_INTEGER frequency;
    QueryPerformanceFrequency(&frequency);

    LARGE_INTEGER starting_time, ending_time, elapsed_microseconds;

    QueryPerformanceCounter(&starting_time);

    //put code to measure speed of here!!!
    Sleep(100);

    QueryPerformanceCounter(&ending_time);
    elapsed_microseconds.QuadPart = ending_time.QuadPart - starting_time.QuadPart;

    //this time is in micro seconds
    auto time_elapsed = static_cast<double>((elapsed_microseconds.QuadPart * 1000000.0) / frequency.QuadPart);

    printf("Time Elapsed: %4.2f\n", time_elapsed);

    printf("All done!\n");

    getchar();

    return 0;
}
```

All this program is going to do is measure a Sleep(100). Keep in mind that Windows is not considered a real time operating system and does not guarantee times like the amount of time a program is told to sleep. But as one can see by the output produced in figure 1, it is pretty close.

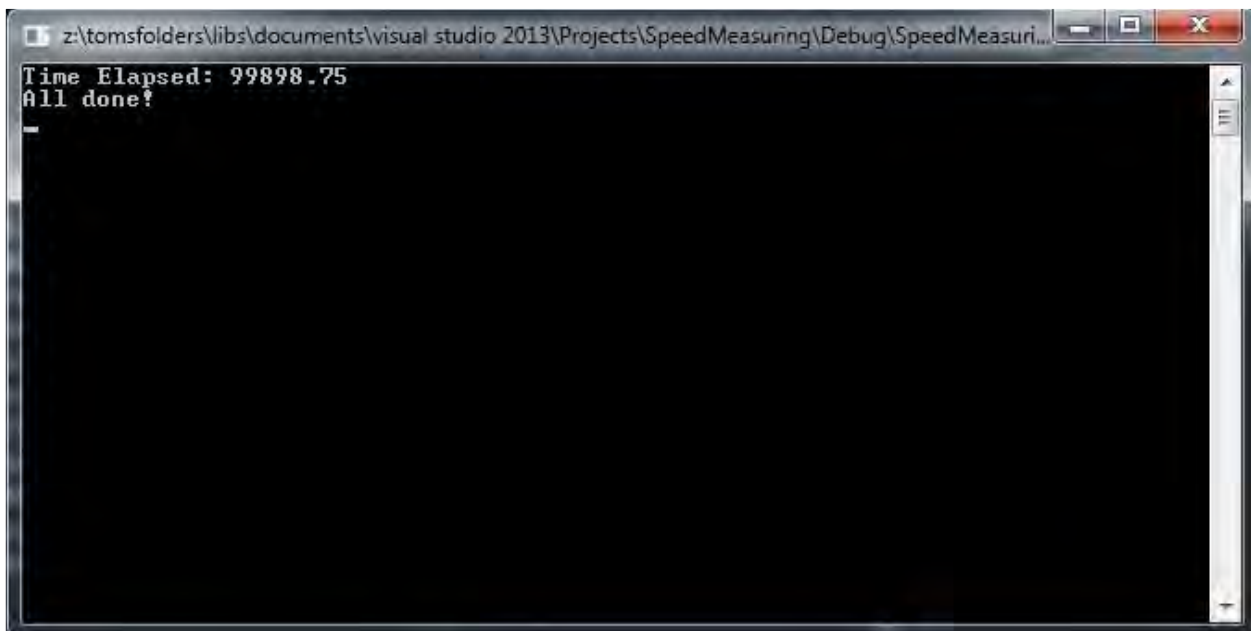


Figure 1  
Output

The program was told to sleep 100 ms and 99.898 ms was measured.



## CONCLUSIONS

Speed measuring in a C++ Windows environment is a relatively easy way for a programmer to gauge the relative efficiency of two comparable sets of code. Using the performance-counters provided in the Windows Application Program Interface, the coder can quickly find performance bottlenecks, measure the impact that a new piece of code will have on an existing code base, or compare the efficiency difference between two code snippets.



UNCLASSIFIED

DISTRIBUTION LIST

U.S. Army ARDEC  
ATTN: RDAR-EIK  
RDAR-WSF-M, T. Nealis  
Picatinny Arsenal, NJ 07806-5000

Defense Technical Information Center (DTIC)  
ATTN: Accessions Division  
8725 John J. Kingman Road, Ste 0944  
Fort Belvoir, VA 22060-6218

GIDEP Operations Center  
P.O. Box 8000  
Corona, CA 91718-8000  
gidep@gidep.org

## REVIEW AND APPROVAL OF ARDEC TECHNICAL REPORTS

Code Speed Measuring for VC++4		_____
Title		Date received by LCSD
Thomas M. Nealis		_____
Author/Project Engineer		Report number (to be assigned by LCSD)
X8048	31	RDAR-WSF-M
Extension	Building	_____
		Author's/Project Engineers Office (Division, Laboratory, Symbol)

## PART 1. Must be signed before the report can be edited.

- a. The draft copy of this report has been reviewed for technical accuracy and is approved for editing.
- b. Use Distribution Statement A, X, B, C, D, E, F or X for the reason checked on the continuation of this form. Reason: Operational Use
1. If Statement A is selected, the report will be released to the National Technical Information Service (NTIS) for sale to the general public. Only unclassified reports whose distribution is not limited or controlled in any way are released to NTIS.
  2. If Statement B, C, D, E, F, or X is selected, the report will be released to the Defense Technical Information Center (DTIC) which will limit distribution according to the conditions indicated in the statement.
- c. The distribution list for this report has been reviewed for accuracy and completeness.

Patricia Alameda

Division Chief

9/9/15  
(Date)

## PART 2. To be signed either when draft report is submitted or after review of reproduction copy.

This report is approved for publication.

Patricia Alameda

Division Chief

9/9/15  
(Date)

Andrew Pskowski

RDAR-CIS

10/15/15  
(Date)

LCSD 49 supersedes SMCAR Form 49, 20 Dec 06